Warsaw University of Technology

Politechnika Warszawska

Distributed Computing and Systems

Magazyn

Author:
Pedro Lindeza
Rafael Belchior

Supervisor:
Wiktor B. Daszczuk

June 24, 2018

POLITECHNIKA

WAR

2

Abstract

Polish consumers are increasingly adhering to online shopping. In their experience of shopping are often faced with alternative stores that offer the same products. For assuring a good experience to the customers, stores must have a system that allows them to effectively manage their products. Furthermore, to allow the possibility of always having stock, the stores must manage effectively their storage (through a warehouse). The Magazyn thus emerges as a product sales helper service that allows stores to manage their products and store them in warehouses. A store has the possibility to search the global inventory of the warehouse and to add products there.

1 Development platform

All project components were implemented on the Java platform, using JDK. The tool to build the project was Maven. Remote invocation of operations is supported by Web Services built with JAX-WS. For the Web Service there is be a server module (Warehouse) and a client module (Store). The Store module contains integration tests to verify the requirements of the respective service, either in normal cases or in cases of error. The tests are be built with the JUnit library. The services must be discovered and located dynamically, through a UDDI server. The services support the simultaneous processing of multiple requests (each request is assigned a execution task), and therefore, we synchronize access to shared variables.

2 Components

2.1 Store

The store is the entity that adds products to a central warehouse.

The Store is initialized with and UDDI URL and wsName or with a wsUrl.

2.2 Warehouse

The warehouse is the storage facility of the stores' products.

Firstly, one SupplierEndpointManager is created, based on its UDDI url, wsName and ws url. After that, the endpoint is created:

```
endpoint = Endpoint.create(this.portImpl);
endpoint.publish(wsURL);
```

. Finally, the warehouse is ready to accept connections:
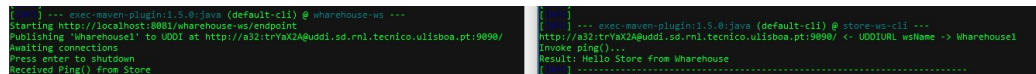
```
System.in.read();
```

.

3 User Guide

- On main directory: *"mvn install -DskipTests"*

- run one instance of the Wharehouse *"cd wharehouse-ws; mvn exec:java"*

- In another console go to store folder *"cd store-ws-cli"*

- To test the connection by ping() *"mvn exec:java"*

- to test *"mvn surefire:test -Dtest=TEST_NAME"*

4 Testing the components

There are two tests related to both of the most finalities of a Warehouse. Search for products or BuyProducts. And a connection test is also implemented.
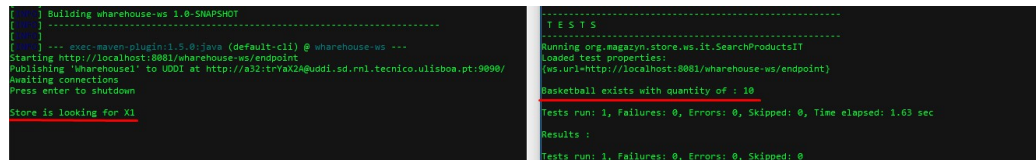
4.1 Ping Warehouse

4.2 Search Product

4.3   Buy Products

```
[INFO] --- exec-maven-plugin:1.5.0:java (default-cli) @ wharehouse-ws ---
Starting http://localhost:8081/wharehouse-ws/endpoint
Publishing 'Wharehouse1' to UDDI at http://a32:trYaX2A@uddi.sd.rnl.tecnico.ulisboa.pt:9090/
Awaiting connections
Press enter to shutdown

Store is looking for X1

ProductID X1 bought 1 time(s)
ProductID Y2 bought 2 time(s)
ProductID Z3 bought 3 time(s)
```

```
-------------------------------------------------
 T E S T S
-------------------------------------------------
Running org.magazyn.store.ws.it.BuyProductIT
Loaded test properties:
{ws.url=http://localhost:8081/wharehouse-ws/endpoint}

Product: X1      Quantity: 10
Product: Y2      Quantity: 20
Product: Z3      Quantity: 30

Buying X1 times 1
Buying Y2 times 2
Buying Z3 times 3

Product: X1      Quantity: 9
Product: Y2      Quantity: 18
Product: Z3      Quantity: 27

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.321 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
```

3